Pose Correction Algorithm for Relative Frames between Keyframes in SLAM

Youngseok Jang[1][0000−0002−6833−8986], Hojoon Shin[2][0000−0002−8437−6018], and H. Jin Kim[3][0000−0002−6819−1136]

[1,2,3] Seoul National University, Seoul 08826, Republic of Korea
[1,2]{duscjs59,asdwer20}@gmail.com [3]hjinkim@snu.ac.kr

Abstract. With the dominance of keyframe-based SLAM in the field of robotics, the relative frame poses between keyframes have typically been sacrificed for a faster algorithm to achieve online applications. However, those approaches can become insufficient for applications that may require refined poses of all frames, not just keyframes which are relatively sparse compared to all input frames. This paper proposes a novel algorithm to correct the relative frames between keyframes after the keyframes have been updated by a back-end optimization process. The correction model is derived using conservation of the measurement constraint between landmarks and the robot pose. The proposed algorithm is designed to be easily integrable to existing keyframe-based SLAM systems while exhibiting robust and accurate performance superior to existing interpolation methods. The algorithm also requires low computational resources and hence has a minimal burden on the whole SLAM pipeline. We provide the evaluation of the proposed pose correction algorithm in comparison to existing interpolation methods in various vector spaces, and our method has demonstrated excellent accuracy in both KITTI and EuRoC datasets.

1 Introduction

Simultaneous localization and mapping (SLAM) has been the focus of numerous research in the field of robotics. SLAM involves estimating the ego-motion of a mobile robot while simultaneously reconstructing the surrounding environment. To this end, visual sensors and laser scanners have been commonly used to perceive the surrounding environment. Vision sensors, in particular, have been most widely adopted due to the wealth of visual information that they can provide at a comparatively low price point. Hence, a vast portion of the SLAM research has been conducted with vision sensors such as monocular, stereo cameras, or RGB-D sensors [1–4].

Operating back-end refinement systems such as pose graph optimization (PGO) or bundle adjustment (BA) on all frames can become taxing especially in

[1,2] These authors contributed equally to this manuscript.
[3] Corresponding author

large-scale environments. To reduce computation time while preserving performance, most modern visual SLAM algorithms adopt keyframe-based approaches which refine only keyframes that contain useful information for SLAM. In other words, keyframe-based SLAM approaches effectively filters the input measurements so that only those that contain significant changes are used in the refinement process, resulting in shorter computation time and local minima avoidance. They allow for the robust estimation of poses and reconstruction of the surrounding map in real-time.

While keyframe-based SLAM methods have dominated SLAM research, they refine the keyframe poses and do not propagate the corrections to the relative frames between keyframes. Because such systems can only make use of selected keyframes that are relatively sparse compared to the raw input measurements, they are not suitable for applications that require corrected poses at high frequency. In particular, multi-robot systems that utilize inter-robot relative poses to integrate multiple observations from team robots require a high robot pose density that existing keyframe-based SLAM methods cannot provide. Therefore, an algorithm that can correct poses of relative frames each time the keyframe poses are updated by the back-end of keyframe-based SLAM is required.

Some attempts to correct the relative poses between keyframes have been made in past works using hierarchical PGO. Hierarchical PGO involves dividing the full pose graph into subgraphs that contain representative keyframes called keynodes. The back-end refinement process is conducted only on these selected keynodes and propagated down the hierarchy. The propagation is usually done through either optimization methods or non-optimization methods such as interpolation. Optimization-based correction methods [5–7] exhibit high accuracy but requires long computation time, making them difficult to operate in real-time. Non-optimization methods [8–10], on the other hand, either treat each subgraph as a rigid body or convert the 3D pose into a vector space and interpolates within the given space, allowing for extremely fast operation. However, in such methods, the interpolation factor can become numerically sensitive when the change in a given axis is small. Furthermore, because the method does not consider the measurement constraints between poses, the correction can potentially break these constraints. These two methods will be further discussed in Section 2.

This paper proposes a pose correction algorithm for relative frames between keyframes. The algorithm requires just the estimated pose output from a SLAM system to operate, meaning that it can be easily integrated into any existing keyframe-based SLAM methods. The generated pose correction also preserves measurement constraints such as image coordinates of visual features without using optimization, enabling fast computation. The proposed algorithm is compared to existing interpolation-based correction methods in various vector spaces and demonstrates superior accuracy and computation time.

The remainder of this paper is structured as follows. The next section reviews related works regarding pose correction and the problem setup and notation are provided in Section 3. Section 4 describes the proposed pose correction algorithm

using measurement constraints. The evaluation results using KITTI and EuRoC datasets are presented in Section 5, and the conclusion of this paper is provided in Section 6.

2 Related Work

This section of the paper discusses the existing work regarding hierarchical PGO and the interpolation in various vector spaces. As mentioned above, distribution of the corrections down to the lower levels of the hierarchy in previous attempts have either used optimization methods or non-optimization methods. This paper will henceforth refer to the former as non-naïve methods and the latter as naïve methods.

The non-naïve approaches to hierarchical PGO involve propagating the refinements of the keyframes to the relative frames through optimization methods. [5, 6] proposed separating the full pose graph into sequentially generated and conditionally independent subgraphs. Pose corrections can be conducted by propagating the error from the most recent subgraph. [7] followed a similar approach but optimized each subgraph independently. While optimizing each hierarchical subgraph is guaranteed to yield accurate results, the procedure requires high computation times and is not suitable for large-scale SLAM applications. Furthermore, because the implementation requires fundamental changes in the SLAM algorithm itself, it is very difficult to integrate such methods into existing keyframe-based SLAM algorithms without affecting the functionality of the algorithms.

The naïve methods simplify the constraints between relative frames to propagate the corrections. Interpolation is the most common example of such simplification methods. However, due to the nature of interpolations, if the rate of change between poses are small, the interpolation factor can become numerically sensitive, resulting in extreme values. Furthermore, the accuracy of the methods also suffers, as the interpolation is only concerned with the keyframe poses and does not consider the measurement constraints present in the pose graph. There have been past attempts to develop alternative methods to interpolation. [8] proposed an algorithm that utilized the quaternion spherical linear interpolation (slerp) algorithm developed in [11] to distribute the pose correction to each frame along the traveled path. However, the method requires the covariance of the edges between nodes and also assumes spherical covariances to compute the interpolation factor. Computing the covariance accurately is very difficult and further assumption of spherical covariance that is not guaranteed SLAM applications can further exacerbate the error. [9] proposed a method where the correction was propagated to the subgraph by treating each subgraph as a rigid body. This simplification assumes that each relative frame receives the same correction and also ignores the measurement constraints between relative frames. More recently, [10] proposed a LiDAR-based online mapping algorithm that treats individual scans as subgraphs and propagates the corrections between scan poses using B-spline interpolation. However, this method discards the mea-

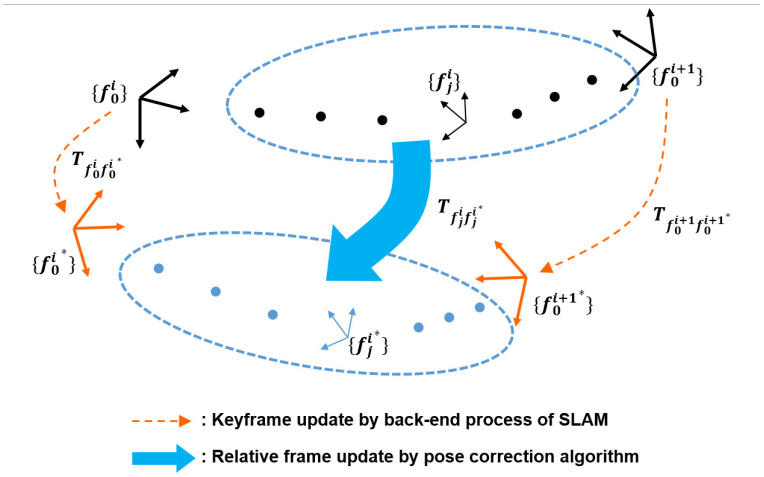

Fig. 1: The refinement process for keyframes and relative frames. The pose correction algorithm for relative frames is triggered each time keyframes are updated by back-end of SLAM.

surement constraints of relative frames between keyframes and does not hold true if the path generated by the robot does not follow a B-spline trajectory.

The interpolation and linearization of various vector spaces that were used in the naïve methods have also been studied extensively [12–15]. Pose corrections require the frame poses to be expressed in SE(3). While the translation component of the SE(3) matrix can be readily interpolated, interpolating the rotation matrix may break the SO(3) constraint that defines the rotation matrix. Hence, the rotation matrix must be converted to a vector space in the form of Euler angles, quaternions, so(3), or rotation axis and angle. In this paper, the numerical robustness of these manifolds was tested for their application in the interpolation of robot poses.

3 Problem Statement

As mentioned previously, most high-performance SLAM algorithms only refine the keyframe poses, which may be too sparse for certain applications. Hence, this paper proposes a fast and easily integrable pose correction algorithm for relative frames between keyframes. The previous approaches have typically used interpolation-based correction methods to achieve fast computation for online robot applications. However, such methods are inherently limited by their numerical sensitivity under singular cases involving small changes in a select axis, and may potentially break measurement constraints even under non-singular conditions. The proposed algorithm is not only capable of preserving measure-

ment constraints under most circumstances, but can also robustly correct poses under singular conditions.

Fig. 1 depicts the correction of relative frames between keyframes when the keyframes have been updated by a refinement process such as PGO or BA. $\{f_0^i\}$ and $\{f_j^i\}$ are the coordinates of the i^{th} keyframe and the j^{th} relative frame connected to the i^{th} keyframe, respectively. The updated keyframe and the corrected relative frame are denoted as $\{f_0^{i^*}\}$ and $\{f_j^{i^*}\}$. $T_{f_1 f_2}$ is the SE(3) transformation from the $\{f_1\}$ coordinate frame to the $\{f_2\}$ coordinate frame. The aim is to approximate the relative frame correction transformation $T_{f_j^i f_j^{i^*}}$ given the keyframe update transformations $T_{f_0^i f_0^{i^*}}$ and $T_{f_0^{i+1} f_0^{i+1^*}}$.

Interpolation approaches are typically used to correct the relative poses between keyframes. However, element-wise interpolation of a matrix in the SE(3) may break the rotation matrix SO(3) constraints $\big(\det(R) = +1 \text{ and } R^{\mathrm{T}} R = I\big)$. To prevent this, the SE(3) matrix should first be converted into a vector. The general equation for the interpolation of an SE(3) matrix after the conversion to a vector is as follows:

$$x_{f_0^{i^*} f_j^{i^*}} = x_{f_0^i f_j^i} + (x_{f_0^{i^*} f_0^{i+1^*}} - x_{f_0^i f_0^{i+1}}) \frac{x_{f_0^i f_j^i}}{x_{f_0^i f_0^{i+1}}} \tag{1}$$

where $x_{f_1 f_2} = f\left(T_{f_1 f_2}\right),\ f : \mathrm{SE}(3) \ \mapsto\ \mathbb{R}^{n \times 1}$, and n is the dimension of the transformed vector space. The above equation was used to interpolate poses in a variety of vector spaces and the results of the interpolation served as the baseline for comparison with the proposed algorithm. The spaces formed by XYZ and the translation portion of the se(3) were used to represent translation while Euler angle, quaternion, and so(3) spaces were used to express rotations. As mentioned above, if even just one component of $x_{f_0^i f_0^{i+1}}$ is small, the resulting interpolation factor becomes numerically sensitive. In particular, if a front-view camera is mounted on a mobile robot or a vehicle, the motion in the z-axis which is the direction of the camera light rays becomes dominant, meaning that the changes in the x and y-axis will be extremely small. Such conditions have a high possibility of resulting in the aforementioned singular case.

4 Pose Correction Algorithm

In this section, the proposed pose correction algorithm is described in detail. The algorithm can be easily integrated into existing keyframe-based SLAM methods as shown in the Fig. 2. Typical SLAM front-end systems estimate the relative pose between the newly acquired image and the most relevant keyframe. Back-end systems select keyframes from the input images and perform graph-based optimization with the keyframes as nodes to improve the keyframe poses. Measurement constraints have been typically used to formulate the likelihood function in optimization methods, but not used in non-optimization methods for faster computation. The aim of the proposed algorithm was to preserve the

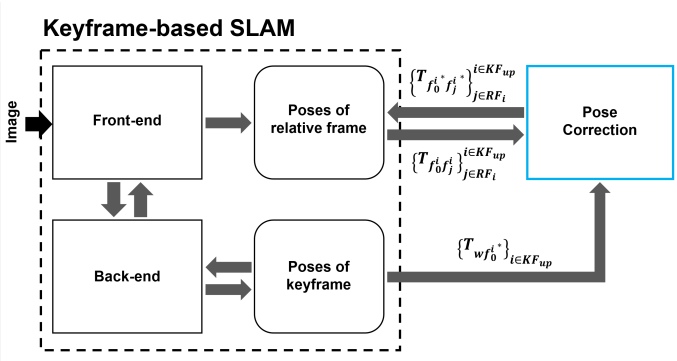

Fig. 2: The overall framework of a keyframe-based SLAM system with the added the pose correction module.

measurement constraints for robustness and accuracy similar to that of the optimization methods, but with a fast computation time similar to that of the non-optimization methods.

The algorithm is triggered when the keyframes are updated and corrects the relative frames connected to the updated keyframes. KF_{up} in Fig. 2 represents the index of the keyframe refined by the back-end and RF_i is the relative frame index connected to the i^{th} keyframe. The correction $T_{f_j^i f_j^{i*}}$ of the relative frames $\{f_j^i\}$ positioned between the i^{th} and $i+1^{\text{th}}$ keyframes can be computed using the keyframe update information $T_{f_0^i f_0^{i*}}$ and $T_{f_0^{i+1} f_0^{i+1*}}$.

4.1 Measurement Constraints

The correction model will now be derived using the measurement constraints. To simplify the notation, we will omit f from the frame notations $\{f_j^i\}$ and show $\{i_j\}$ when the frame notations are used as subscripts or superscripts. Equations (2) and (3) show the projection equation of $\{f_0^i\}$ and $\{f_0^{i*}\}$ respectively.

$$\{f_0^i\}: \quad {}^{i_0}P_k = {}^{i_0}\lambda_k K^{-1}\, {}^{i_0}\bar{u}_k \tag{2}$$

$$\{f_0^{i*}\}: \quad {}^{i_0{}^*}P_k^* = {}^{i_0{}^*}\lambda_k^* K^{-1}\, {}^{i_0{}^*}\bar{u}_k \tag{3}$$

$$\text{where} \quad {}^{i_0}P_k = {}^{i_0}\begin{bmatrix} X \\ Y \\ Z \end{bmatrix}_k, \quad {}^{i_0}\bar{u}_k = {}^{i_0}\begin{bmatrix} u \\ v \\ 1 \end{bmatrix}_k.$$

Here, ${}^{i_0}P_k$, ${}^{i_0}\bar{u}_k$ and ${}^{i_0}\lambda_k$ are the 3-D position, the homogeneous pixel coordinates, and depth of the k^{th} feature with respect to $\{f_0^i\}$, respectively. K is the intrinsic parameter matrix, and the position of the updated landmark ${}^{i_0{}^*}P_k$ can

be expressed using (2) and (3) as follows:

$$\begin{aligned} {}^{i_0{}^*}P_k^* &= R_{i_0{}^*i_0}\left({}^{i_0}P_k + {}^{i_0}\delta P_k\right) + t_{i_0{}^*i_0} \\ &\approx \frac{{}^{i_0{}^*}\lambda_k^*}{{}^{i_0}\lambda_k}\,{}^{i_0}P_k \quad \left(\because \ {}^{i_0}\bar{u}_k = {}^{i_0{}^*}\bar{u}_k\right) \end{aligned} \tag{4}$$

where $R_{i_0{}^*i_0}$ and $t_{i_0{}^*i_0}$ are the rotation and translation from $\{f_0^{i^*}\}$ to $\{f_0^i\}$, respectively. ${}^{i_0}\delta P_k$ is the variation of the k^{th} landmark position by the refinement process of SLAM. (4) was derived using the condition that the measurement of the landmark remains constant regardless of the pose corrections. The projection equation of $\{f_j^i\}$ and $\{f_j^{i^*}\}$ with respect to the updated landmark ${}^{i_0{}^*}P_k^*$ shown in (4) can now be expressed as follows:

$$\{f_j^i\} : \frac{{}^{i_0}\lambda_k}{{}^{i_0{}^*}\lambda_k^*}\,R_{i_ji_0}\,{}^{i_0{}^*}P_k^* + t_{i_ji_0} = {}^{i_j}\lambda_k K^{-1\,i_j}\bar{u}_k \tag{5}$$

$$\{f_j^{i^*}\} : R_{i_j{}^*i_0{}^*}\,{}^{i_0{}^*}P_k^* + t_{i_j{}^*i_0{}^*} = {}^{i_j{}^*}\lambda_k^* K^{-1\,i_j{}^*}\bar{u}_k \tag{6}$$

$$\begin{aligned} {}^{i_0{}^*}P_k^* &= \frac{{}^{i_0{}^*}\lambda_k^{*\,i_j}\lambda_k}{{}^{i_0}\lambda_k}\,R_{i_0i_j}K^{-1\,i_j}\bar{u}_k + \frac{{}^{i_0{}^*}\lambda_k^*}{{}^{i_0}\lambda_k}\,t_{i_0i_j} && (7) \\ &= {}^{i_j{}^*}\lambda_k^*\,R_{i_0{}^*i_j{}^*}K^{-1\,i_j{}^*}\bar{u}_k + t_{i_0{}^*i_j{}^*}\ . && (8) \end{aligned}$$

Using the fact that the measurement ${}^{i_j}\bar{u}_k$ observed in each image remains constant regardless of the update, (9) can be derived from (7) and (8).

$$\left(\frac{{}^{i_0{}^*}\lambda_k^{*\,i_j}\lambda_k}{{}^{i_j{}^*}\lambda_k^{*\,i_0}\lambda_k}R_{i_0i_j} - R_{i_0{}^*i_j{}^*}\right){}^{i_j{}^*}\lambda_k^* K^{-1\,i_j{}^*}\bar{u}_k + \frac{{}^{i_0{}^*}\lambda_k^*}{{}^{i_0}\lambda_k}t_{i_0i_j} - t_{i_0{}^*i_j{}^*} = 0\ . \tag{9}$$

The depth value of each feature increases as the translational difference between the keyframes in which the features were observed increases. Using this characteristic and assuming that the translation ratio and the depth ratio are equal, the following condition is derived:

$$s_i = \frac{{}^{i_0{}^*}\lambda_k^*}{{}^{i_0}\lambda_k} \approx \frac{\left\|t_{i_0{}^*(i+1)_0{}^*}\right\|_2^2}{\left\|t_{i_0(i+1)_0}\right\|_2^2}, \quad \frac{{}^{i_0{}^*}\lambda_k^{*\,i_j}\lambda_k}{{}^{i_j{}^*}\lambda_k^{*\,i_0}\lambda_k} \approx 1\ . \tag{10}$$

Applying the (10) to (9) yields an identical equation (11) for $\left({}^{i_j{}^*}\lambda_k^* K^{-1\,i_j{}^*}\bar{u}_k\right)$. For the identical equation to hold for all measurements, the solution must be expressed as in (12).

$$\left(R_{i_0i_j} - R_{i_0{}^*i_j{}^*}\right){}^{i_j{}^*}\lambda_k^* K^{-1\,i_j{}^*}\bar{u}_k + s_i\ t_{i_0i_j} - t_{i_0{}^*i_j{}^*} = 0 \tag{11}$$

$$R_{i_0{}^*i_j{}^*} = R_{i_0i_j}, \quad t_{i_0{}^*i_j{}^*} = s_i\ t_{i_0i_j}\ . \tag{12}$$

(12) was derived using the measurement constraint between the i^{th} keyframe and $\{f_j^i\}$. Applying the same procedure to the $i+1^{\text{th}}$ keyframe yields (13).

$$R_{(i+1)_0{}^*i_j{}^*} = R_{(i+1)_0i_j}, \quad t_{(i+1)_0{}^*i_j{}^*} = s_i\ t_{(i+1)_0i_j}\ . \tag{13}$$

4.2 Fusion with Two Constraints

By fusing the conditions (12) and (13) derived previously, $T_{i_0 i_j{}^*}$ can now be computed. The gap between the solutions to the aforementioned conditions can be expressed as follows:

$$\begin{aligned}\delta R &= {R^{KF_i}_{i_0{}^* i_j{}^*}}^T R_{i_0{}^*(i+1)_0{}^*} R^{KF_{i+1}}_{(i+1)_0{}^* i_j{}^*} \\ &= {R_{i_0 i_j}}^T R_{i_0{}^*(i+1)_0{}^*} R_{(i+1)_0 i_j} \\ \delta t &= t^{KF_i}_{i_0{}^* i_j{}^*} + R_{i_j{}^* i_0{}^*} t_{i_0{}^*(i+1)_0{}^*} + R_{i_j{}^*(i+1)_0{}^*} t^{KF_{i+1}}_{(i+1)_0{}^* i_j{}^*} \\ &= R_{i_j{}^* i_0{}^*} \left(t_{i_0{}^*(i+1)_0{}^*} - s_i (t_{i_0 i_j} - R_{i_0{}^*(i+1)_0{}^*} t_{(i+1)_0 i_0}) \right)\end{aligned} \tag{14}$$

where $R^{KF_i}_{i_0{}^* i_j{}^*}$ and $t^{KF_i}_{i_0{}^* i_j{}^*}$ are the corrected relative rotation and translation computed from (12), and $R^{KF_{i+1}}_{i_0{}^* i_j{}^*}$ and $t^{KF_{i+1}}_{i_0{}^* i_j{}^*}$ are the correction terms computed from (13). The δR and δt terms in (14) are expressed with respect to $\{f_j^{i*}\}$, which is estimated under the conditions given in (12). To compensate for the gap, fusion as expressed in (16) is performed to estimate the corrected relative frame.

$$R_{i_0{}^* i_j{}^*} = R^{KF_i}_{i_0{}^* i_j{}^*} \cdot \mathrm{SLERP}\left(\delta R,\ \alpha_j^i\right) \tag{15}$$

$$t_{i_0{}^* i_j{}^*} = t^{KF_i}_{i_0{}^* i_j{}^*} + \mathrm{LERP}\left(R_{i_0{}^* i_j{}^*} \delta t,\ \alpha_j^i\right) \tag{16}$$

where SLERP$(\cdot)$ and LERP$(\cdot)$ are spherical linear interpolation and linear interpolation functions, respectively. δR is converted to a quaternion space to be utilized in the SLERP function. α_j^i is the interpolation factor which should reflect the reliability of conditions given by (12) and (13). Since the number of reliable edges increases as the distance between frames decreases due to the increase in the number of shared features, the ratio of the distance from $\{f_j^i\}$ to $\{f_0^i\}$ and the distance from $\{f_j^i\}$ to $\{f_0^{i+1}\}$ was used as the interpolation factor in this paper.

5 Experimental Result

This section provides the results of the proposed pose correction algorithm integrated with ORB-SLAM2 [1] which is one of the most popular keyframe-based SLAM. As mentioned above, the existing interpolation-based methods were tested in various vector spaces to function as a baseline for comparison. The interpolations for translation were done in XYZ and the translation component v of the se(3) spaces, while the rotation components were interpolated in Euler angles, quaternion, and so(3) spaces. The stereo images of KITTI [16] and EuRoC [17] benchmarks datasets were used for analysis. ORB-SLAM2 was used to generate the poses of frames, though any appropriate keyframe-based SLAM can be applied.

There are two types of refinements that occur in the back-end of SLAM: local BA and global BA. Local BA occurs when a new keyframe is added and refines

only keyframes that have a strong connection to the newly added keyframe. Global BA occurs when a loop is detected and refines all keyframes that are in the map. The proposed pose correction module is triggered whenever local or global BA takes place and uses the updated keyframes and their relative frames as inputs. We analyze the accuracy of the corrected relative frame poses and the computation time required to compute the correction. However, because the keyframe poses computed by the SLAM system inherently contain error, the difference between the corrected relative frames poses and the ground truth (GT) may not purely reflect the correction performance. Therefore, an additional post processing step was introduced to the SLAM system to directly evaluate the correction performance of the algorithm. The poses of keyframes that lie on the estimated trajectory by ORB-SLAM2 with the correction module was additionally updated to their GT poses so that the keyframes now lie on the GT. The proposed algorithm was used to correct the relative frames so that the final output of the SLAM is corrected to the GT. The difference between these final corrected relative frame poses and the GT poses was used as the error metric for correction. Tests were performed on a laptop (Y520-15IKBN, 16GB RAM with Intel i7-7700HQ @ 2.80GHz $\times$ 4cores).

5.1 KITTI Dataset

The KITTI dataset [16] is generated from a stereo camera mounted on top of a vehicle, where the yaw motion and the camera z-axis movement are dominant with minimal motion along other axis due to the characteristics of a vehicle platform.

The ORB-SLAM2 and correction algorithm results obtained from the sequences (00-10) of the KITTI dataset are summarized in table 1 and 2 respectively. No-correction in table 2 refers to the results obtained from the simple concatenation of relative frames and keyframes without correction. The proposed algorithm outperformed all the baseline interpolation methods in all sequences except for sequence 01. As can be seen in table 1, almost all image frames became keyframes using ORB-SLAM2 in sequence 01, meaning that the correction module had a minimal effect. For translation, the proposed algorithm nearly doubled the mean accuracy of the baseline method in sequences 00, 02, 03, 07 and 08. Furthermore, the algorithm yielded a lower standard deviation when compared to the baseline methods. Since standard deviation indicates the robustness of the system in a variety of situations, it can be concluded that the proposed algorithm is not numerically sensitive compared to the baseline methods. Rotation, on the other hand, does not exhibit significant differences in accuracy between methods. This is because the KITTI dataset was acquired using a ground vehicle, resulting in very little rotation aside from yaw. There are, however, significant differences in the standard deviation for rotation, meaning that singular cases occur in certain areas of the sequences, resulting in significant error. In some sequences, especially for rotations, the baseline methods yielded worse results than the no-correction method. Because rotation has such a small error even prior to correction, the numerical error has a significant effect on the results.

The resultant trajectory from each translation space for the select segments A and B in sequence 00 is shown in figure 3. Segment A visualizes the varying performance of the baseline methods, as both the XYZ and v interpolations stray wildly from the GT poses, even more so than the no-correction method. The proposed method, however, was able to remain consistent with the GT poses throughout the entire segment. Segment B shows the singular case in v, resulting in a huge deviation away from the GT. It is worth noting that the singularity occurred only in the v space interpolation and not the XYZ space interpolation. This is a clear depiction of the numerical sensitivity of the existing baseline methods, as such deviations can result in large error that may be even worse than the cases without correction at all. The proposed algorithm, however, performed robustly in both cases, demonstrating the improved accuracy and numerical robustness the algorithm has over the baseline methods.

Table 1: The results of ORB-SLAM2 in KITTI dataset. The number of keyframes and all frames and loop closures are indicated as shown.

KITTI	00	01	02	03	04	05	06	07	08	09	10
# of Keyframes	1355	1047	1742	226	155	717	473	251	1199	588	321
# of All Frames	4541	1101	4661	801	271	2701	1101	1101	4071	1504	1201
Loop	O	X	O	X	X	O	O	O	X	X	X

Table 2: The summary of results for the KITTI dataset. The blue indicates the lowest error while red indicates the highest error. Each cell contains the mean $\pm$ standard deviation, and (median). v represents the translation component of se(3) space.

	Translation (cm)				Rotation ($\times 10^{-1}$deg)				
Seq.	No-Correction	XYZ	v	Proposed	No-Correction	Euler	Quat	so(3)	Proposed
00	2.034±1.76	1.919±3.91	2.949±9.84	0.947±0.79	0.618±0.59	0.891±1.37	0.954±1.60	0.955±1.60	0.473±0.35
	(1.425)	(0.984)	(1.037)	(0.698)	(0.445)	(0.472)	(0.473)	(0.473)	(0.378)
01	1.874±0.52	0.885±0.46	1.083±0.84	0.915±0.46	0.190±0.08	0.190±0.08	0.191±0.09	0.191±0.09	0.188±0.08
	(1.890)	(0.788)	(0.876)	(0.833)	(0.188)	(0.188)	(0.189)	(0.189)	(0.188)
02	1.737±1.13	1.535±1.90	1.786±2.37	0.916±0.61	0.442±0.31	0.561±0.56	0.591±0.67	0.592±0.67	0.392±0.25
	(1.458)	(0.993)	(1.05)	(0.762)	(0.362)	(0.377)	(0.382)	(0.383)	(0.327)
03	1.455±0.87	1.610±2.07	1.692±2.10	0.775±0.47	0.478±0.23	0.594±0.53	0.558±0.40	0.559±0.40	0.456±0.21
	(1.228)	(0.985)	(1.002)	(0.665)	(0.403)	(0.417)	(0.417)	(0.417)	(0.393)
04	1.045±0.50	0.798±0.50	0.818±0.52	0.726±0.42	0.267±0.15	0.346±0.32	0.346±0.32	0.346±0.32	0.254±0.14
	(1.021)	(0.701)	(0.731)	(0.658)	(0.023)	(0.242)	(0.240)	(0.239)	(0.225)
05	1.495±1.32	1.008±0.83	1.166±1.31	0.849±0.63	0.469±0.34	0.672±0.85	0.746±1.14	0.746±1.14	0.405±0.28
	(1.074)	(0.747)	(0.757)	(0.656)	(0.375)	(0.420)	(0.422)	(0.422)	(0.328)
06	1.204±1.06	0.813±0.81	0.814±0.80	0.639±0.43	0.339±0.25	0.711±1.78	0.658±1.47	0.723±1.97	0.283±0.17
	(0.855)	(0.591)	(0.585)	(0.509)	(0.258)	(0.292)	(0.292)	(0.295)	(0.248)
07	2.034±2.17	2.648±4.74	2.784±5.02	1.372±2.12	0.537±0.42	0.717±1.03	0.679±0.84	0.679±0.84	0.418±0.28
	(1.267)	(0.875)	(0.886)	(0.701)	(0.428)	(0.423)	(0.423)	(0.423)	(0.342)
08	2.153±2.01	1.464±1.72	1.571±1.89	0.883±0.62	0.481±0.35	0.658±0.74	0.759±1.13	0.762±1.14	0.394±0.26
	(1.49)	(0.925)	(0.957)	(0.714)	(0.394)	(0.420)	(0.425)	(0.425)	(0.335)
09	1.362±0.82	0.997±0.70	1.179±1.12	0.868±0.52	0.415±0.27	0.499±0.44	0.527±0.53	0.527±0.53	0.381±0.24
	(1.167)	(0.819)	(0.875)	(0.755)	(0.367)	(0.378)	(0.382)	(0.383)	(0.334)
10	1.484±1.22	1.078±0.98	1.443±1.77	0.806±0.62	0.502±0.32	0.719±0.71	0.730±0.76	0.730±0.76	0.438±0.28
	(1.108)	(0.788)	(0.854)	(0.626)	(0.447)	(0.536)	(0.536)	(0.537)	(0.386)

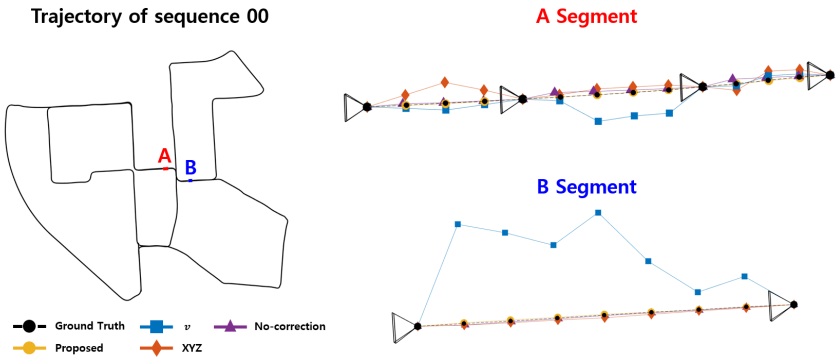

Fig. 3: The translational results of the correction algorithm for two select segments in sequence 00 of KITTI.

5.2 EuRoC Dataset

The EuRoC dataset [17] is generated from a stereo camera mounted on a micro aerial vehicle (MAV) and contains a more diverse range of motion compared to the KITTI data. The MAV was flown in an industrial environment (machine room) and two different rooms with a motion capture system in place. There are a total of 11 sequences, with each sequence classified as easy, medium, and difficult depending on the motion of the MAV and the room environment. Since ORB-SLAM2 does not provide sufficient results for V2_03_difficult due to significant motion blur in some of the frames, the particular sequences were not used in this paper.

The results of the ORB-SLAM2 and correction algorithm obtained from the remaining ten sequences are described in tables 3 and 4, respectively. The EuRoC dataset is generated in a much smaller environment than the KITTI dataset, resulting in sparser keyframes. Furthermore, because the motion of a MAV is erratic compared to a ground vehicle, the pose error from SLAM is more significant than the KITTI dataset. Hence, the need for a correction algorithm is more apparent. The proposed algorithm demonstrates significantly improved results for both translation and rotation and the improvements are clearer than the KITTI dataset. In sequence V102, for example, the standard deviation of the algorithm was almost four times lower than that of the second best method for translation (no-correction) and five times lower than that of the second best method for rotation (no-correction), demonstrating its robustness. The mean and median values have also been halved, showcasing the accuracy of the algorithm.

The corrected trajectory for select segments of sequence V102 is shown in figure 4. Unlike the KITTI dataset, the no-correction method becomes meaningless, as the concatenation of relative frames and keyframes results in discontinuous trajectories as shown in both segments A and B. Furthermore, in both segments, the baseline methods failed to remain consistent with the GT, showing

significant deviations in particular around the second keyframe in segment A. In segment B, the numerical sensitivity of the v space interpolation method causes the corrected poses to deviate significantly from the GT trajectory. The XYZ space interpolation method is also unable to achieve the desired correction and results in significant clustering around the midpoint between the two keyframes. The proposed method, on the other hand, was able to remain close to the GT trajectory with no singularities. Hence, even under erratic motion that causes the baseline methods to fail, the proposed algorithm was still able to generate accurate and robust corrections.

Table 3: The number of keyframes and all frames within the EuRoC dataset. The sequences with loop closures are indicated.

EuRoC	Machine Hall					Vicon Room				
	01	02	03	04	05	101	102	103	201	202
# of Keyframes	483	431	436	302	353	109	151	208	216	271
# of All Frames	3638	2999	2662	1976	2221	2871	1670	2093	2148	2309
Loop	X	X	X	X	O	X	X	O	X	X

Table 4: The summary of results for the EuRoC dataset. The conventions are same as in table 2.

	Translation (cm)				Rotation (deg)				
Seq.	No-Correction	XYZ	v	Proposed	No-Correction	Euler	Quat	so(3)	Proposed
MH01	8.130±7.53	4.385±9.58	5.026±10.84	1.937±2.28	5.904±7.69	4.014±5.14	3.827±4.68	3.840±4.70	3.214±4.78
	(4.817)	(1.076)	(1.219)	(1.076)	(2.095)	(1.453)	(1.529)	(1.532)	(1.110)
MH02	7.592±7.90	3.028±5.03	2.947±4.87	1.974±2.75	2.212±2.70	2.456±4.20	2.632±4.73	2.654±4.79	1.672±2.17
	(4.783)	(0.913)	(0.977)	(0.766)	(1.174)	(0.724)	(0.706)	(0.706)	(0.759)
MH03	22.693±24.15	11.755±16.50	15.135±24.25	5.047±6.31	4.455±5.20	3.687±5.12	4.028±6.64	4.247±7.55	3.236±3.79
	(15.429)	(4.596)	(5.254)	(2.244)	(2.852)	(1.502)	(1.479)	(1.496)	(1.374)
MH04	15.201±15.92	6.229±10.01	6.427±9.93	2.994±4.88	2.605±2.81	2.301±3.64	2.239±3.53	2.221±3.49	1.260±1.34
	(10.145)	(1.260)	(1.468)	(0.913)	(1.669)	(0.747)	(0.735)	(0.735)	(0.772)
MH05	13.922±15.47	5.928±15.00	4.847±9.92	1.753±2.47	2.376±3.08	1.660±2.59	1.885±3.14	1.951±3.34	0.969±1.27
	(9.143)	(1.374)	(1.497)	(0.686)	(1.090)	(0.598)	(0.595)	(0.597)	(0.443)
V101	25.256±27.57	23.694±38.42	24.674±32.08	9.682±13.43	17.598±20.90	13.402±19.10	17.400±25.00	17.844±25.86	5.732±9.19
	(16.314)	(8.886)	(10.596)	(5.475)	(9.802)	(5.417)	(6.230)	(6.230)	(3.008)
V102	28.821±33.85	24.904±45.06	26.179±39.96	6.548±9.65	22.823±38.48	24.537±43.96	20.791±39.68	20.973±39.66	4.841±5.74
	(15.540)	(4.680)	(7.292)	(2.651)	(9.039)	(5.196)	(5.332)	(5.428)	(2.475)
V103	17.162±19.95	15.725±30.72	24.530±52.15	6.410±9.15	13.051±13.45	10.602±14.66	8.516±12.07	8.441±11.94	6.457±9.22
	(9.932)	(3.755)	(4.757)	(2.027)	(8.663)	(4.226)	(3.584)	(3.560)	(2.674)
V201	5.090±4.36	2.771±4.57	2.617±3.53	1.488±1.72	3.784±4.22	4.486±7.77	5.421±10.84	5.664±11.64	1.473±1.63
	(3.877)	(1.110)	(1.217)	(0.747)	(2.375)	(1.400)	(1.401)	(1.407)	(0.878)
V202	11.916±11.99	11.250±22.75	10.486±20.86	4.308±5.47	8.694±9.44	7.329±10.264	8.677±14.22	8.760±14.42	3.539±4.07
	(8.187)	(2.764)	(2.841)	(1.890)	(5.637)	(3.300)	(3.062)	(3.071)	(1.883)

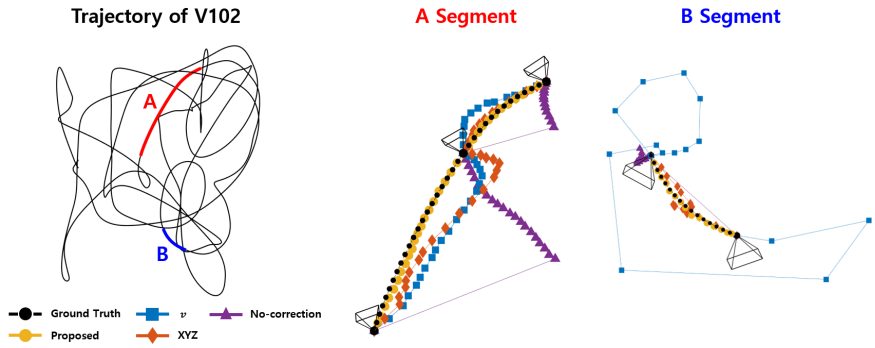

Fig. 4: The results of the corrected translation for two select segments in V102 of EuRoC.

5.3 Computation time

The proposed correction module triggers whenever keyframe refinement, or in other words BA, occurs in keyframe-based SLAM. Table 5 shows the amount of time required for the algorithm to compute a single correction in a MATLAB environment. Although there were no significant differences between the proposed and baseline algorithms for translation, the proposed algorithm required the most time for rotation. This is because the SLERP algorithm used in the proposed algorithm requires more computations than a simple interpolation approach. However, the difference in the median computation time is approximately 1 millisecond and the algorithm only runs when a new keyframe is selected or when a loop is closed, meaning that the computation time required is insignificant when compared to the entire SLAM pipeline. In addition, the standard deviation of the computation time is large compared to the median value. This is due to the presence of loops within certain sequences, which results in a global BA. As discussed previously, global BA updates all keyframes, meaning that all relative frames must be corrected. Hence, typical computation time for a typical local BA is similar to that of the median value.

Table 5: The computation time taken for a single correction operation. The conventions are same as in table 2.

Translation (msec)			Rotation (msec)			
XYZ	v	Proposed	Euler	Quat	so(3)	Proposed
0.340±0.86 (0.170)	0.698±1.74 (0.313)	0.356±0.89 (0.135)	1.781±4.66 (0.702)	2.205±5.66 (0.944)	0.689±1.69 (0.318)	3.916±9.87 (1.441)

6 Conclusion

In this paper, we have proposed a lightweight pose correction algorithm for relative frames between keyframes that can be easily integrated into existing keyframe-based SLAM systems. The algorithm was derived by preserving the measurement constraints of two updated keyframes and utilizing the notion that the measurement observed in both keyframes remains constant regardless of the update. By doing so, the algorithm avoids singularities and numerical sensitivity that existing interpolation-based methods suffer from. The algorithm was applied to poses generated from the current state-of-the-art ORB-SLAM2 in KITTI and EuRoC datasets. The algorithm demonstrated results superior to the existing interpolation methods in both translation and rotation for all three datasets. The computation time of the proposed algorithm was only a few milliseconds longer than the baseline methods, which is negligible in the overall SLAM process. Applications requiring visual information that may appear in non-keyframes can benefit from the proposed algorithm with negligible cost to computation time. Since the proposed module can be easily attached to existing keyframe-based SLAM systems, the algorithm may be used in a wide range of fields.

References

1. Mur-Artal, R., Tardós, J.D.: Orb-slam2: An open-source slam system for monocular, stereo, and rgb-d cameras. IEEE Transactions on Robotics **33** (2017) 1255–1262
2. Engel, J., Koltun, V., Cremers, D.: Direct sparse odometry. IEEE Transactions on Pattern Analysis and Machine intelligence **40** (2017) 611–625
3. Forster, C., Zhang, Z., Gassner, M., Werlberger, M., Scaramuzza, D.: Svo: Semidirect visual odometry for monocular and multicamera systems. IEEE Transactions on Robotics **33** (2016) 249–265
4. Engel, J., Schöps, T., Cremers, D.: Lsd-slam: Large-scale direct monocular slam. In: European Conference on Computer Vision, Springer (2014) 834–849
5. Piniés, P., Tardós, J.D.: Scalable slam building conditionally independent local maps. In: 2007 IEEE/RSJ International Conference on Intelligent Robots and Systems, IEEE (2007) 3466–3471
6. Piniés, P., Tardós, J.D.: Large-scale slam building conditionally independent local maps: Application to monocular vision. IEEE Transactions on Robotics **24** (2008) 1094–1106
7. Suger, B., Tipaldi, G.D., Spinello, L., Burgard, W.: An approach to solving large-scale slam problems with a small memory footprint. In: 2014 IEEE International Conference on Robotics and Automation (ICRA), IEEE (2014) 3632–3637
8. Grisetti, G., Grzonka, S., Stachniss, C., Pfaff, P., Burgard, W.: Efficient estimation of accurate maximum likelihood maps in 3d. In: 2007 IEEE/RSJ International Conference on Intelligent Robots and Systems, IEEE (2007) 3472–3478
9. Grisetti, G., Kümmerle, R., Stachniss, C., Frese, U., Hertzberg, C.: Hierarchical optimization on manifolds for online 2d and 3d mapping. In: 2010 IEEE International Conference on Robotics and Automation, IEEE (2010) 273–278

10. Droeschel, D., Behnke, S.: Efficient continuous-time slam for 3d lidar-based online mapping. In: 2018 IEEE International Conference on Robotics and Automation (ICRA), IEEE (2018) 1–9
11. Shoemake, K.: Animating rotation with quaternion curves. In: Proceedings of the 12th annual Conference on Computer Graphics and Interactive Techniques. (1985) 245–254
12. Stuelpnagel, J.: On the parametrization of the three-dimensional rotation group. SIAM Review **6** (1964) 422–430
13. Shuster, M.D., et al.: A survey of attitude representations. Navigation **8** (1993) 439–517
14. Barfoot, T., Forbes, J.R., Furgale, P.T.: Pose estimation using linearized rotations and quaternion algebra. Acta Astronautica **68** (2011) 101–112
15. Blanco, J.L.: A tutorial on se(3) transformation parameterizations and on-manifold optimization. University of Malaga, Tech. Rep **3** (2010)
16. Geiger, A., Lenz, P., Stiller, C., Urtasun, R.: Vision meets robotics: The kitti dataset. The International Journal of Robotics Research **32** (2013) 1231–1237
17. Burri, M., Nikolic, J., Gohl, P., Schneider, T., Rehder, J., Omari, S., Achtelik, M.W., Siegwart, R.: The euroc micro aerial vehicle datasets. The International Journal of Robotics Research **35** (2016) 1157–1163